Parallel in time Integration of Dynamo Simulations

Andrew Clarke
Supervised by: Daniel Ruprecht, Chris Davies, Steven Tobias

University of Leeds

September 2, 2019

This work was undertaken on ARC3, part of the High Performance Computing facilities at the University of Leeds, UK.

Why Study Dynamos?

Generate magnetic field on Earth, Sun.
Lots of unexplained phenomena - Sunspots, Geomagnetic Reversals, prediction of Geomagnetic field decline, 11 year maunder period

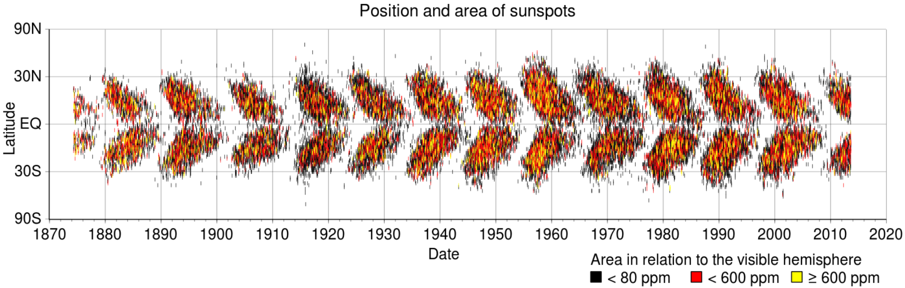


Figure 1: Maunder Butterfly sunspots plot

What makes up a Dynamo

Convective flow of conducting fluids.
Dynamo action - induction equation.
Navier Stokes + Maxwell equations.

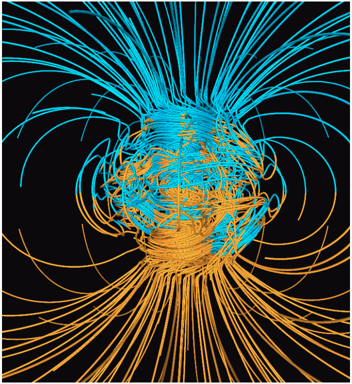


Figure 2: Simulation of Earth Magnetic Field. Glatzmaiers and Roberts, 1995[1]

[1] Glatzmaier and Roberts, "Computer simulation of a geomagnetic field reversal".

Numerical Simulations

Dynamo problem is difficult to simulate.

- Stiff nonlilnear equations.
- Huge range of time and length scales.

Simulations ran in very unrealistic parameter regimes.

Parameter	Earth	Simulations
Ekman	10^{-15}	$10^{-7} - 10^{-3}$
Rayleigh	10^{24}	$10^{6} - 10^{9}$
Magnetic Prantl	10^{-6}	$0.1 - 10$
Prandtl	10^{-2}	$10^{-1} - 1$
Magnetic Reynolds	10^{3}	$40 - 3000$
Reynolds	10^{9}	$10 - 5000$

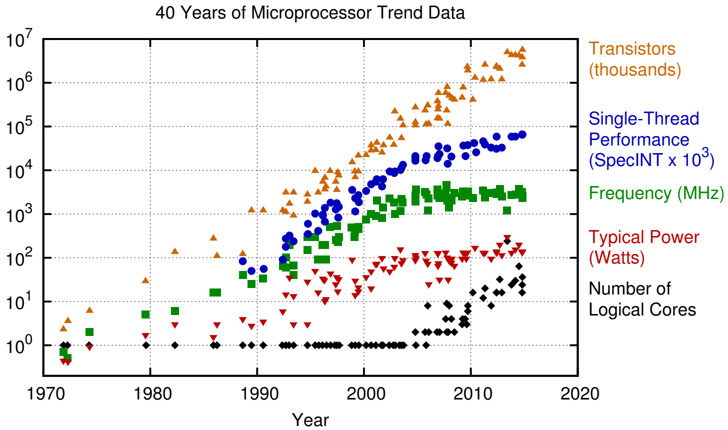
Table 1: Roberts and King, 2013[2]/ Schaeffer, 2017[3]

[2] Roberts and King, "On the genesis of the Earth's magnetism".

[3] Schaeffer et al., "Turbulent geodynamo simulations: a leap towards Earth's core".

Motivation

To gain more speed, we need to utilise more processors.
Traditional codes are parallel in space, but reach a scaling limit.



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2015 by K. Rupp

Parallel in Time

- Extra direction for parallelization.
- Works with existing spatial parallelization.
- Compute the start of the simulation at the same time as the end of the simulation*.

Figure 3: https://parallel-in-time.org/

Parareal Algorithm: Lions et al, 2001[4]

$$U_{n+1}^{k+1} = \underbrace{\mathcal{G}_{\Delta t}(U_n^{k+1})}_{\text{Coarse integrator}} + \underbrace{\mathcal{F}_{\delta t}(U_n^k)}_{\text{Fine integrator}} - \underbrace{\mathcal{G}_{\Delta t}(U_n^k)}_{\text{Coarse integrator}} \tag{1}$$

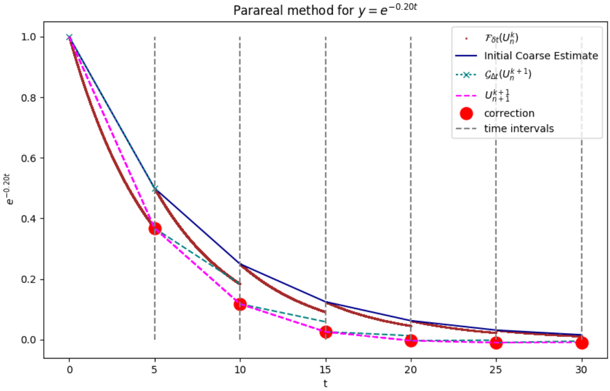
Parareal Solver:

- Usual solver $\mathcal{F}$ with usual time step δt
- Parallelize in time by using an iterative approach, which combines $\mathcal{F}$ with an approximation $\mathcal{G}$.
- $\mathcal{G}$ is called the coarse solver, and usually has a bigger timestep Δt.
- Solution U found at iteration $k+1$ using the algorithm above.

[4] Lions, Maday, and Turinici, "Résolution d'EDP par un schéma en temps pararéel".

Parareal Example

$$U_{n+1}^{k+1} = \underbrace{\mathcal{G}_{\Delta t}(U_n^{k+1})}_{\text{Coarse integrator}} + \underbrace{\mathcal{F}_{\delta t}(U_n^{k})}_{\text{Fine integrator}} - \underbrace{\mathcal{G}_{\Delta t}(U_n^{k})}_{\text{Coarse integrator}} \tag{2}$$



Theoretical speed up:

$$s = \left[\left(1 + \frac{\text{Iterations}}{\text{processors}} \right) \frac{R_c}{R_f} + \frac{\text{Iterations}}{\text{processors}} \right]^{-1} \tag{3}$$

where R_c is coarse runtime and R_f is the fine runtime over one time slice. Speed up is bounded by

$$s \leq \min \left\{ \frac{\text{Processors}}{\text{Iterations}}, \frac{R_f}{R_c} \right\}, \tag{4}$$

Rule of thumb: if you have R_c just less than R_f, low number of iterations, if $R_c <<< R_f$, then high number of iterations.

Parareal Example

Very fast coarse solver - lots of iterations, less time for each iteration

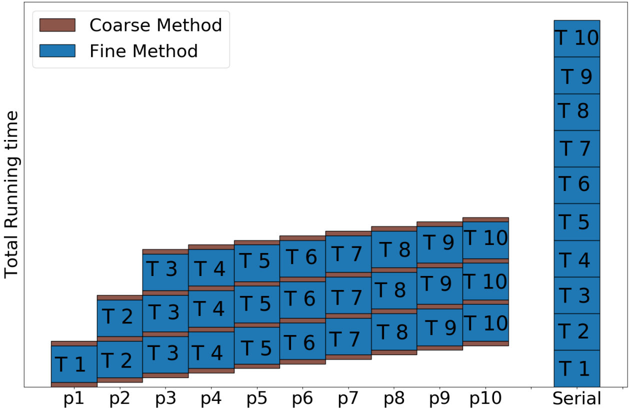


Figure 4: Time spent for each processor (10 processors, R_F/R_C large)

Parareal Example

More accurate coarse solver - less iteration, but more time required for each iteration

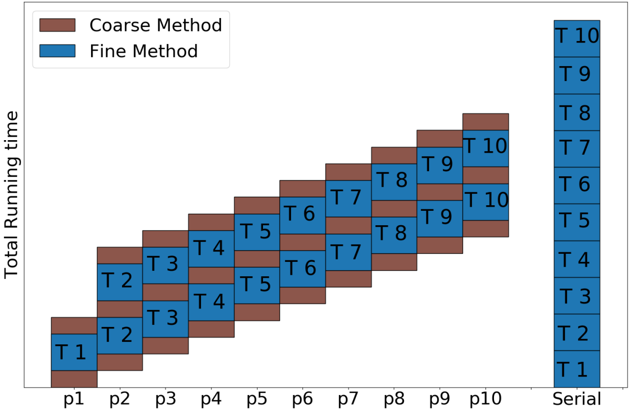


Figure 5: Time spent for each processor (10 processors, R_F/R_C large)

Numerics/ Computing Details

Spatial Discretisation:

- Pseudospectral collocation method.
- Fourier series in x and z (kinematic), Chebyshev polynomials in z (Rayleigh-Bénard)

Dedalus Spectral Solver[5]

- Open source python solver.
- Optimized parallel libraries: FFTW, ATLAS/MKL, MPI

Timestepping

- Implicit-Explicit Runge-Kutta
- Linear/ high order terms treated implicitly
- Non-linear terms treated explicitly

[5] Burns et al., "Dedalus: A Flexible Framework for Numerical Simulations with Spectral Methods".

Dynamo Problem

Full (Boussinesq) magnetohydrodynamic problem

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\rho_0^{-1}\nabla p + \alpha \boldsymbol{g} T + \boldsymbol{j} \times \boldsymbol{B} + \nu \nabla^2 \boldsymbol{u} \tag{5}$$

$$\frac{\partial T}{\partial t} + \boldsymbol{u} \cdot \nabla T = \kappa \nabla^2 T \tag{6}$$

$$\frac{\partial \mathbf{B}}{\partial t} = \nabla \times (\mathbf{u} \times \mathbf{B}) + \eta \nabla^2 \mathbf{B}. \tag{7}$$

$$\nabla \cdot \boldsymbol{u} = 0 \tag{8}$$

$$\nabla \cdot \boldsymbol{B} = 0 \tag{9}$$

Simplification - ignore magnetic field.

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\rho_0^{-1}\nabla p + \alpha \boldsymbol{g} T + \xcancel{\boldsymbol{j} \times \boldsymbol{B}} + \nu \nabla^2 \boldsymbol{u} \tag{10}$$

$$\frac{\partial T}{\partial t} + \boldsymbol{u} \cdot \nabla T = \kappa \nabla^2 T \tag{11}$$

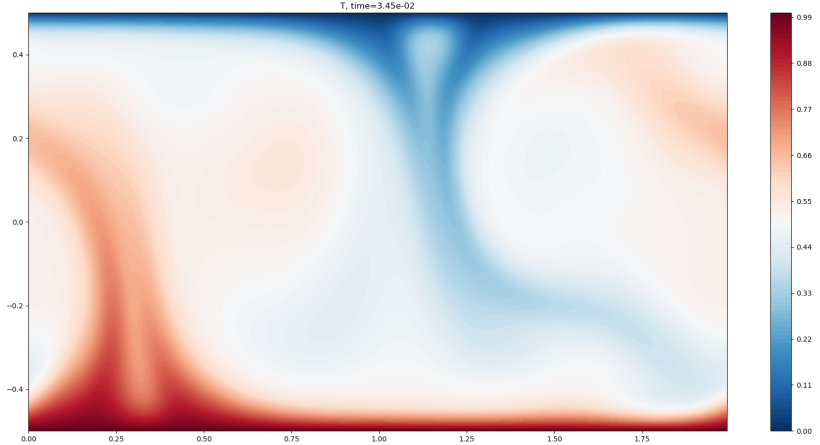
$$\xcancel{\frac{\partial \mathbf{B}}{\partial t} = \nabla \times (\mathbf{u} \times \mathbf{B}) + \eta \nabla^2 \mathbf{B}} \tag{12}$$

$$\nabla \cdot \boldsymbol{u} = 0 \tag{13}$$

$$\xcancel{\nabla \cdot \boldsymbol{B} = 0} \tag{14}$$

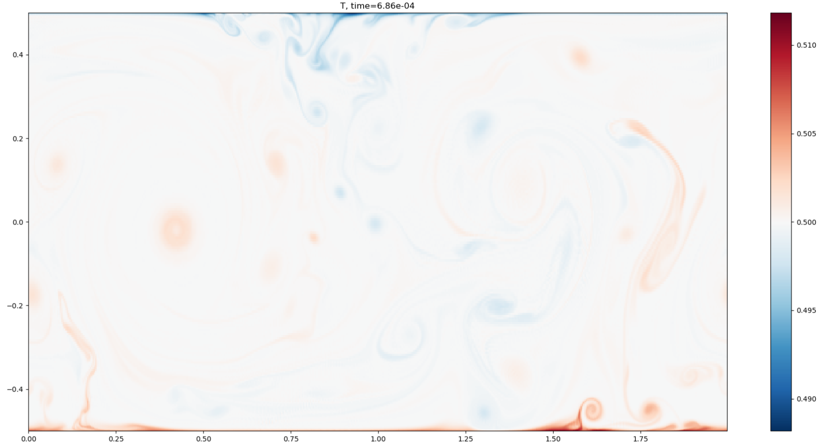
Rayleigh Bénard Convection

Rayleigh $= 10^6$, Prandtl $= 1$



Rayleigh Bénard Convection

Rayleigh $= 10^9$, Prandtl $= 1$



Rayleigh-Bénard: Setting Fine Solver

Nusselt number:

- Calculate at multiple z planes.
- Max relative defect should be below 1%, Mound and Davies, 2017[6].

Energy balance:

- Compare buoyancy production with viscous dissipation.
- Check that they match within 1%, Mound and Davies, 2017[8].

Time Averages:

- Average over at least 100 turnover times - until statistically steady.

Boundary Layers:

- 6 points: Verzicco and Camussi (2003)[7], 7 points: Stevens (2010)[8].

---

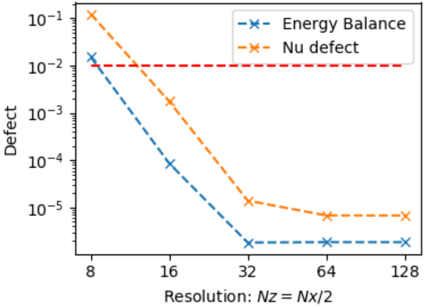
[6] Mound and Davies, “Heat transfer in rapidly rotating convection with heterogeneous thermal boundary conditions”.

[7] Verzicco and Camussi, “Numerical experiments on strongly turbulent thermal convection in a slender cylindrical cell”.

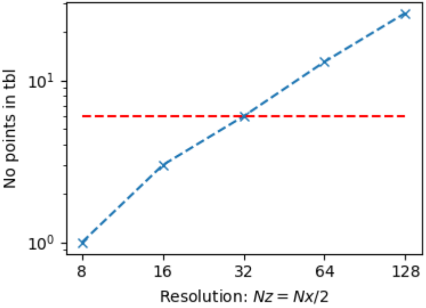
[8] Stevens, Verzicco, and Lohse, “Radial boundary layer structure and Nusselt number in Rayleigh–Bénard convection”.

Rayleigh-Bénard: Resolutions

Convergence of solution with space.
Red lines indicate the threshold required of solution.
Fine solution set to 32(Nz) x 64(Nx)



(a) Nusselt and Energy Balance Error



(b) Number of points in TBL.

Rayleigh-Bénard: Parareal Results 1

Clear difference between coarse/fine data.

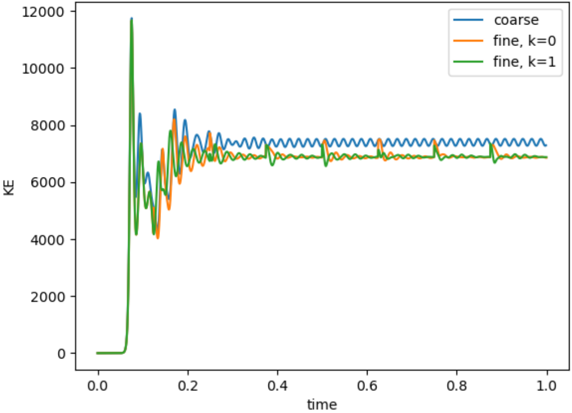


Figure 7: How time series data looks for increasing iteration number.

Rayleigh-Bénard: Parareal Convergence

Parareal convergence of simulations. (8 time slices)

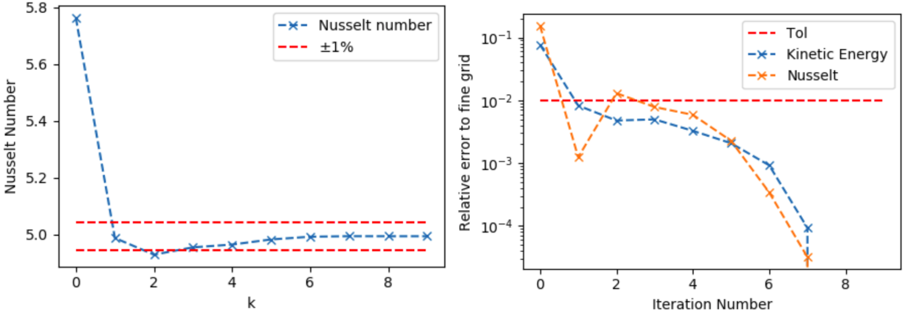
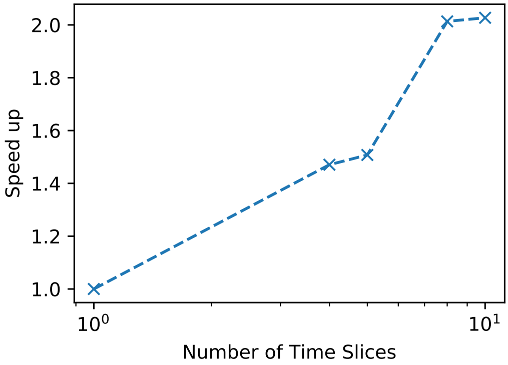


Figure 8: Time averaged values compared with high resolution study

Scaling Results:
$Ra = 10^5$, $Pr = 1$, Fine Res $= (64 \times 32)$, coarsening factor $= 4$.

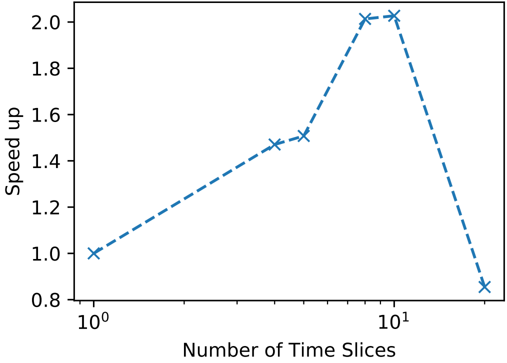
$$\frac{\text{coarse dt}}{\text{fine dt}} = 4$$



Rayleigh-Bénard: Parareal Scaling

Scaling Results:
$Ra = 10^5$, $Pr = 1$, Fine Res = (64×32), coarsening factor = 4.

$$\frac{\text{coarse dt}}{\text{fine dt}} = 4$$



Kinematic Dynamo

Prescribe $\boldsymbol{u}$

$$\cancel{\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\rho_0^{-1}\nabla p + \alpha \boldsymbol{g} T + \boldsymbol{j} \times \boldsymbol{B} + \nu \nabla^2 \boldsymbol{u}} \tag{15}$$

$$\cancel{\frac{\partial T}{\partial t} + \boldsymbol{u} \cdot \nabla T = \kappa \nabla^2 T} \tag{16}$$

$$\frac{\partial \mathbf{B}}{\partial t} = \nabla \times (\mathbf{u} \times \mathbf{B}) + \eta \nabla^2 \mathbf{B}. \tag{17}$$

$$\nabla \cdot \boldsymbol{u} = 0 \tag{18}$$

$$\nabla \cdot \boldsymbol{B} = 0 \tag{19}$$

$$\boldsymbol{u} = \boldsymbol{u}(t, \boldsymbol{x}) \tag{20}$$

Kinematic Dynamo

- Subset of full dynamo problem - prescribed flow
- Non-dimensionalised Induction Equation

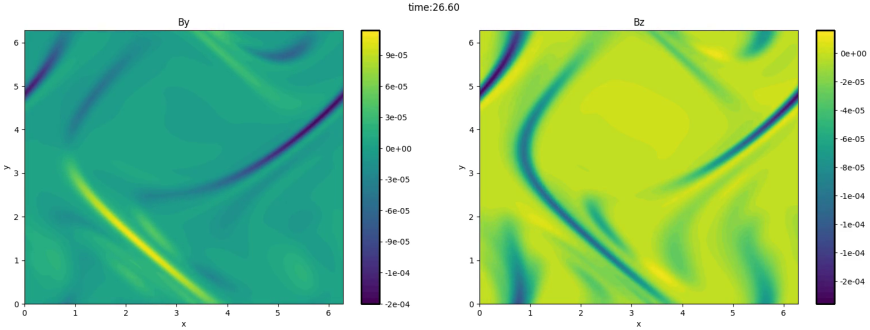
$$\frac{\partial \boldsymbol{B}}{\partial t} = \boldsymbol{\nabla} \times (\boldsymbol{u} \times \boldsymbol{B}) + \frac{1}{R_m} \nabla^2 \boldsymbol{B} \tag{21}$$

- $R_m = UL/\eta$ is the magnetic Reynolds number. Characteristic Length scale L, characteristic velocity U, magnetic diffusivity η, characteristic time $\tau = L/U$.
- Pre-defined velocity field $\boldsymbol{u} = (u_x, u_y, u_z)$, unknown magnetic field
- Divergence free

$$\nabla \cdot \boldsymbol{B} = \nabla \cdot \boldsymbol{u} = 0$$

Galloway Proctor Dynamo

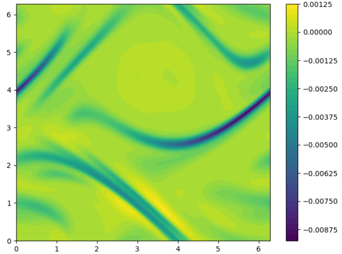
Magnetic Reynolds = 300



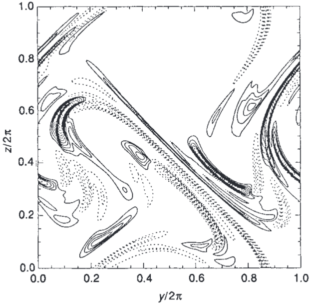
Galloway Proctor[9]

Time dependant opposing cylindrical flow. Periodic in y and z.

$$\boldsymbol{u} = \sin(z + \sin\omega t) + \cos(y + \cos\omega t), \cos(z + \sin\omega t), \sin(y + \cos\omega t). \tag{22}$$



(a) Contours of By

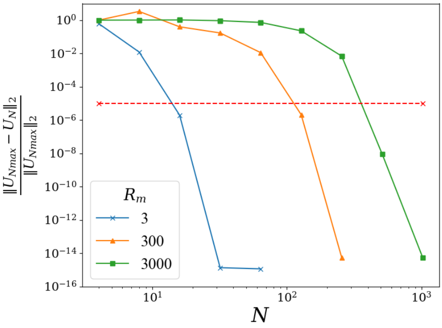


(b) Contours of Bx

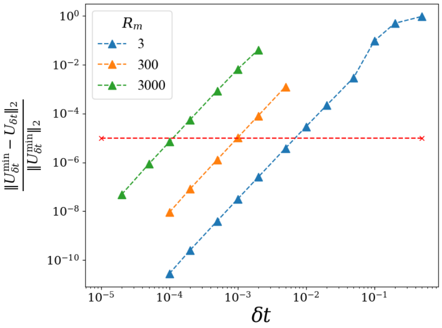
[9] Galloway and Proctor, "Numerical calculations of fast dynamos in smooth velocity fields with realistic diffusion".

Fine solver - Kinematic Dynamo

Fine Spatial resolution $N_y = N_z = N_F$.
Time step δt



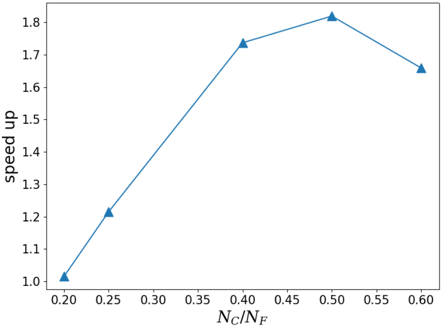
(a) Spatial convergence for different R_m



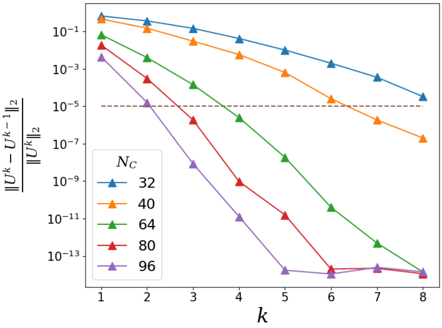
(b) Temporal convergence for different R_m

Figure 10: Galloway Proctor Growth Convergence

Coarse Spatial resolution $N_y = N_z = N_C$.
Time step increased in line with stability.



(a) Speed up

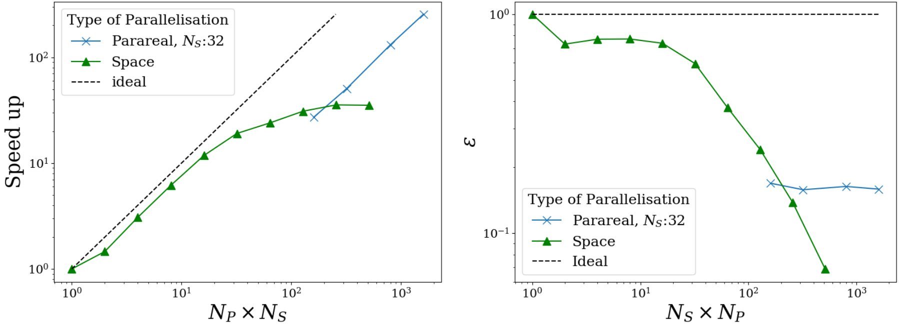


(b) Parareal convergence

Figure 11: Parareal Convergence for different Coarse resolutions ($N_F = 160$)

Results - Kinematic Dynamo

Coarse time step much larger than fine time step, creates a larger difference in computational complexity.
This gives better opportunity for overall speed up.

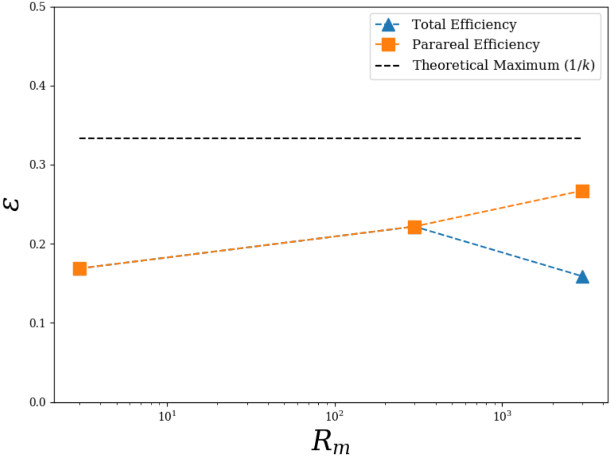


(a) Speed up (b) Efficiency

Figure 12: Galloway-Proctor flow, $R_m = 3000$. Total number of processers against speed up/ efficiency

Performance With changing R_m

- At $R_m = 3$ and 300, we only parallelise using parareal.
- At $R_m = 3000$, we parallelise in time and space, so total efficiency and parareal efficiency are different.



Thank you for listening

References:

Clarke, A. T., Davies, C. J., Ruprecht, D., & Tobias, S. M. (2019). Parallel-in-time integration of Kinematic Dynamos. *arXiv preprint arXiv:1902.00387.*

Future work:

- Further Study of Rayleigh-Bénard Convection
- pySDC + Dedalus, to implement PFASST parallel in time algorithm
- Release Parareal solver add-on for Dedalus to "work" with any equation.